<u>In the claims:</u>

1.   (Currently Amended)   A method for implementing a pre-designed state model, said method comprising:

extracting state information from the state model <u>via at least one controller</u>;

processing said extracted state information <u>via said at least one controller</u>;

generating a state code and a state table in response to said processed extracted state information <u>via said at least one controller</u>;

compiling said state code to generate a runtime code <u>via said at least one controller</u>; and then

implementing the state model by running said runtime code while utilizing information within said state table using a ~~separate~~ <u>designated</u> controller <u>that is separate from said at least one controller</u>.

2.   (Original)   A method as in claim 1 wherein extracting state information from the state model comprises determining what events exist in the state model.

3.   (Original)   A method as in claim 1 wherein extracting state information from the state model comprises determining what transitions exist between states within the state model.

4.   (Original)   A method as in claim 1 further comprising:

generating an events symbols header in response to a header file; and

generating said state code in response to said processed extracted state information and said events symbols header.

5.   (Original)   A method as in claim 4 wherein compiling said state code comprises compiling said state code in response to said events symbols header.

6.   (Original)   A method as in claim 1 further comprising:

generating a events symbols header in response to an events configuration file; and

generating said state code in response to said processed extracted state information and said events symbols header.

7.   (Previously Presented)   A method as in claim 1 further comprising annotating the state model using a script language to alter state behavior.

8.   (Currently Amended)   A method for implementing a pre-designed plurality of state models for a state machine having an event configuration file, said method comprising:

extracting state information from the plurality of state models;

generating an events symbols header having global and shared event symbol definitions from the event configuration file, the event configuration file and said events symbol header defining events to be shared;

processing said extracted state information in response to said events symbols header;

generating a plurality of state codes and a plurality of state tables in response to said processed extracted state information;

compiling said plurality of state codes using said events symbols header to generate a plurality of runtime codes; and

implementing the state models by running said plurality of runtime codes while referring to said plurality of state tables.

9.   (Original)   A method as in claim 8 wherein implementing a pre-designed plurality of state models comprises implementing a cooperating set of run-time controllers.

10.   (Previously Presented)   A method as in claim 8 further comprising:

generating said events symbols header in response to a header file; and

generating said plurality of state codes in response to said processed extracted state information and said events symbols header.

11.   (Currently Amended)   A state processor for generating a state table and a runtime code for use in implementing one or more pre-designed state models, said device comprising:

a state model information provider extracting state model information in response to the one or more state models having hierarchial organized states;

a state information separator generating a state code and the state table in response to the one or more state models; and

a compiler compiling said state code and generating the runtime code.

12. (Original) A device as in claim 11 further comprising:

an event organizer generating an event symbols header in response to a header file; and

said compiler compiling said state code using said event symbols header.

13. (Original) A device as in claim 12 wherein said event organizer generates an event symbols header comprising a centralized list of all events for adding or renaming events.

14. (Original) A device as in claim 12 wherein said event symbols header comprises global and shared event symbol definitions.

15. (Original) A device as in claim 12 wherein said header file comprises global and shared event symbol definitions.

16. (Previously Presented) A device as in claim 11 further comprising a runtime library comprising:

at least one event processor; and

an interpreter.

17.   (Original)   A device as in claim 16 wherein said runtime library comprises a generic state machine component for implementing of event handling.

18.   (Previously Presented)   A device as in claim 16 wherein said interpreter is a time and memory efficient interpreter for processing and handling events.

19.   (Previously Presented)   A device as in claim 16 wherein said at least one event processor comprises a scripted dynamic events processor for annotating the one or more state models to alter state behavior.

20.   (Previously Presented)   A device as in claim 11 wherein said state processor generates a plurality of state tables and a plurality of state codes in response to the one or more state models.